
# YCSB++: Benchmarking Cloud DBs

## HEC FSIO 2011, Arlington VA

August 10, 2011

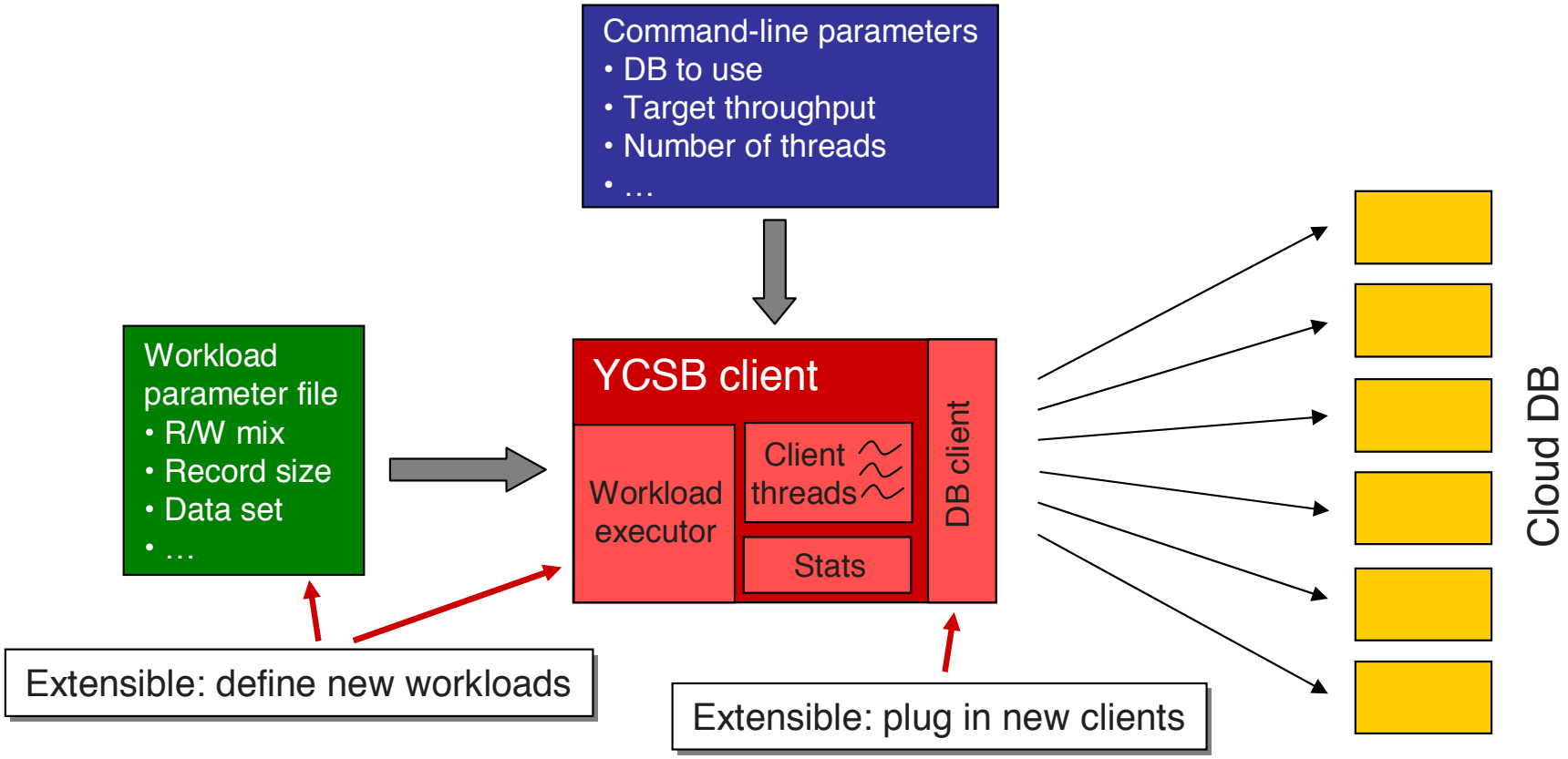
Garth Gibson, Julio Lopez, Swapnil Patil, Milo Polte, Kai Ren,
Wittawat Tantisiriroj, Lin Xiao

to appear in SOCC 2011

Carnegie Mellon
Parallel Data Laboratory

# Extending a Prior Benchmark Tool

- Yahoo! Cloud Serving Benchmark (YCSB) tool
  - steady state load of CRUD (create-read-update-delete) operations



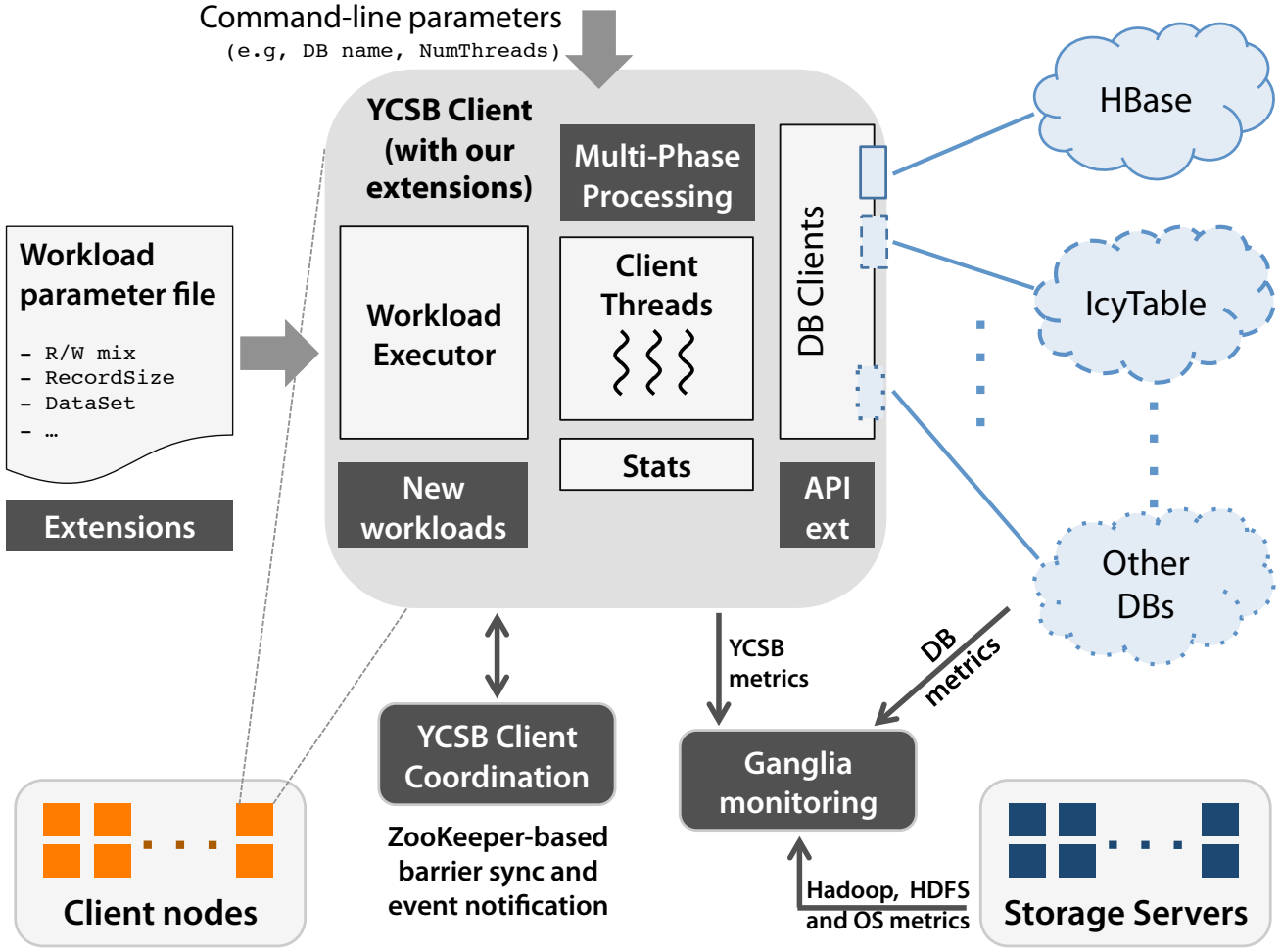



github.com/brianfrankcooper/YCSB [SOCC 2010]

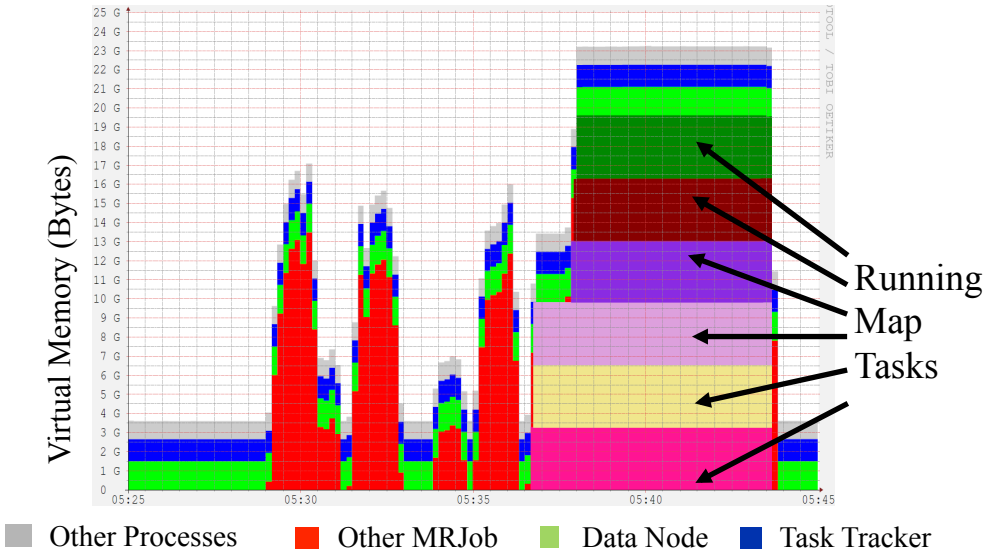
# Adv. Features of YCSB++

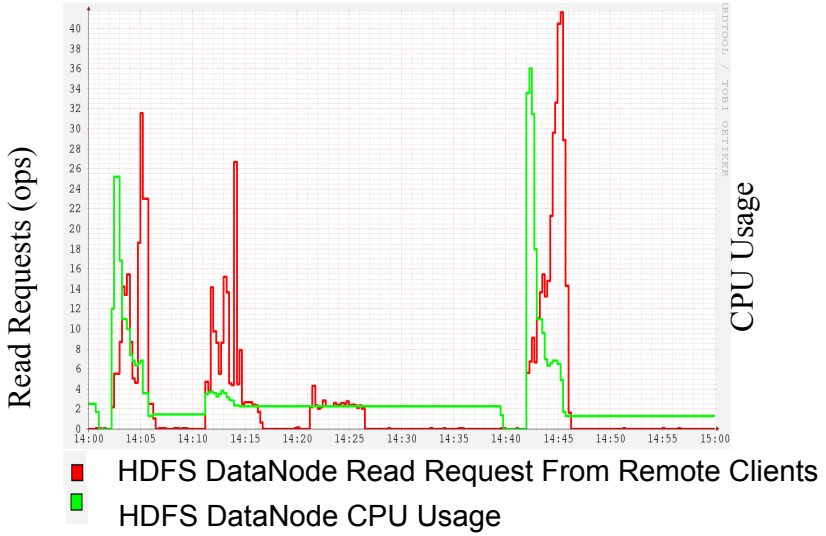
- High Ingest Rate Features
  - Deep batch writing
  - Pre-splitting tablets (given future insert distribution)
  - Bulk-load: MR format map files externally
- Read Features
  - Read-after-write: what price eventual consistency?
  - Offloading filtering to servers
  - Security ACLs – what performance price?
- Better interpretation of monitoring
  - Integrate knowledge of services, user jobs (Otus)

# YCSB++ Framework

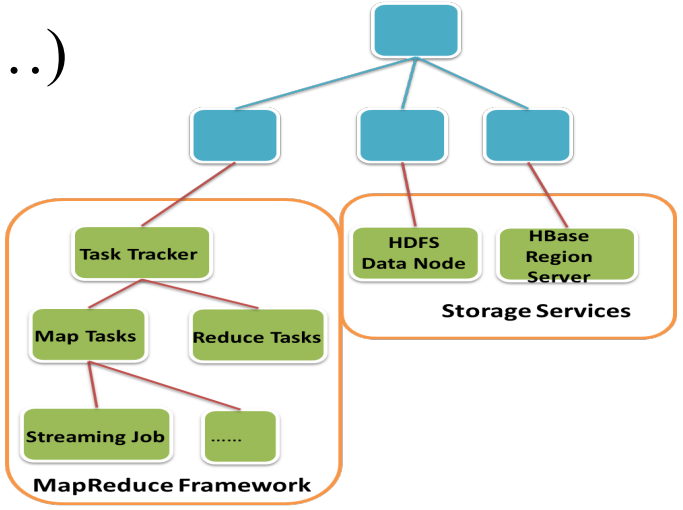


# Extensions for Monitoring (Otus)

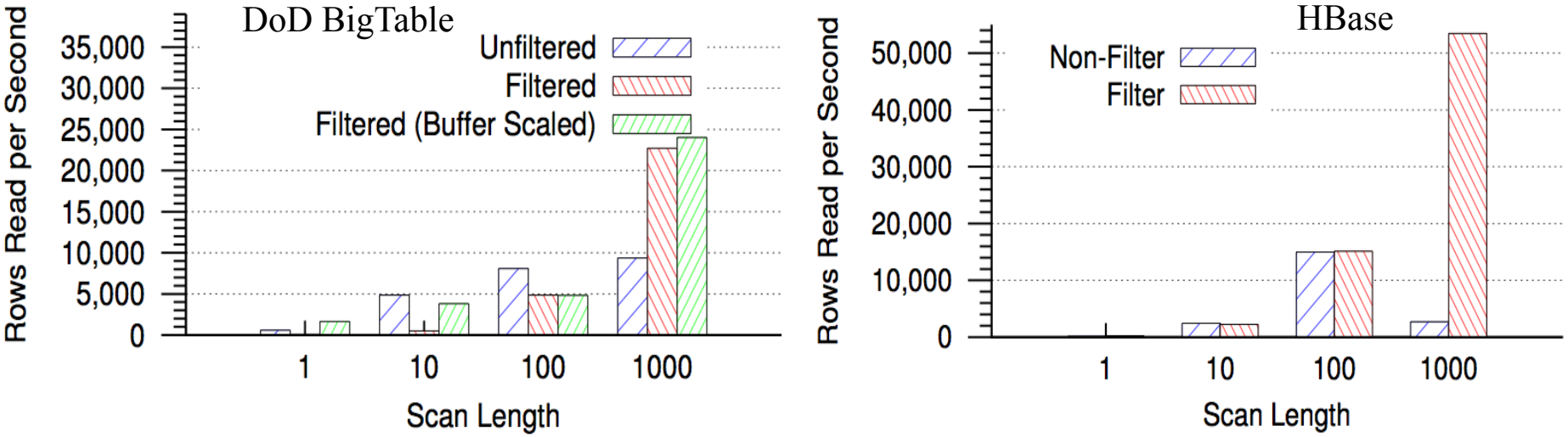



- Service stats (Hadoop, Hbase, HDFS, …)
- Walk process group tree looking for specific command lines
  - Aggregate stats for subgroups
- Customizable displays

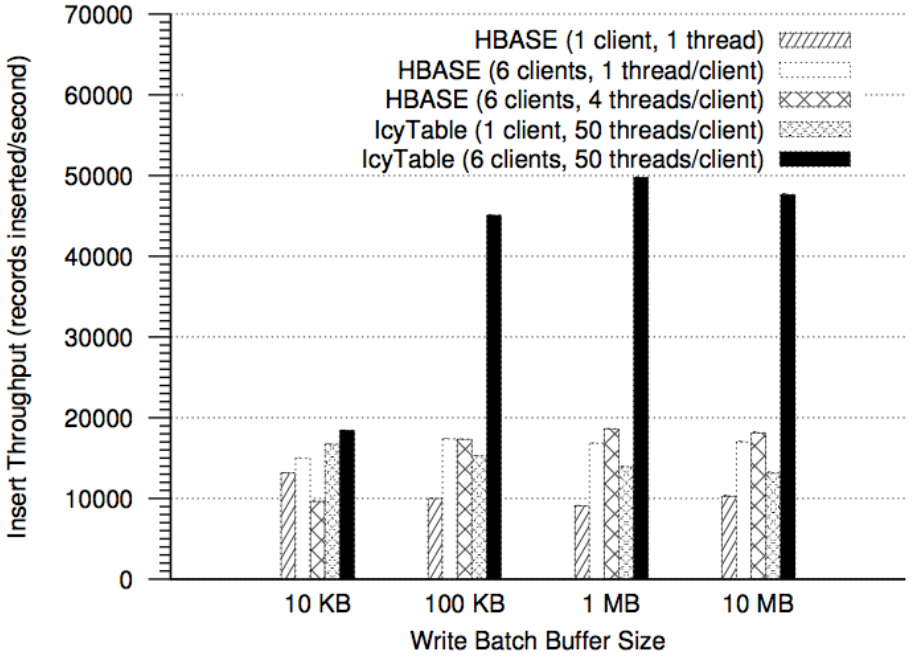


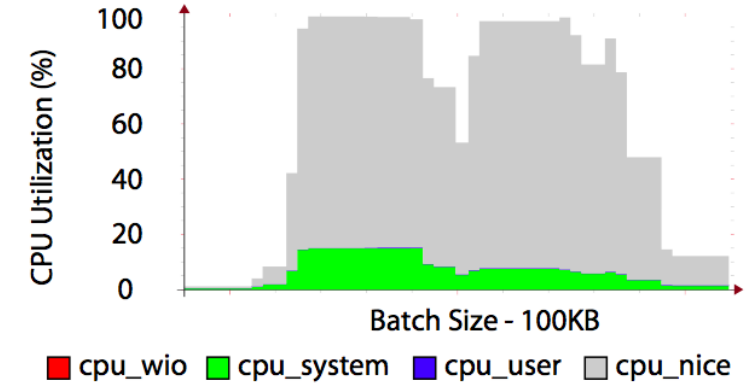
# Server side Filtering

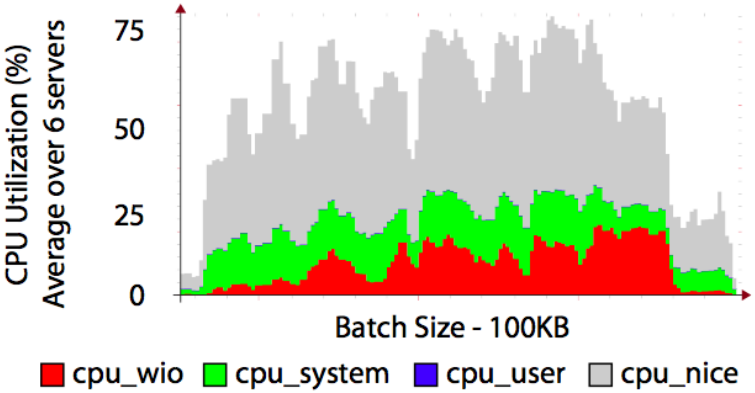



- Filtering when little data is desired leads to excessive prefetching on the server, because it fills the scanner batch
  - Size the scanner batch to the expected result size (scaled buffer)
- Hbase table was decomposed into more columnar stores, so DoD BigTable does more work

# Batch writers & eventual consistency

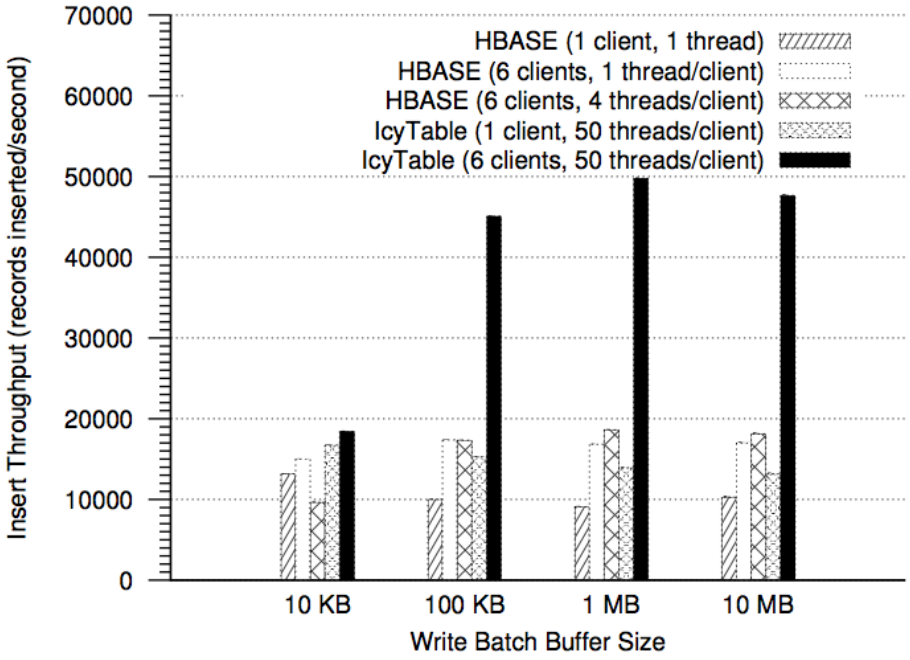


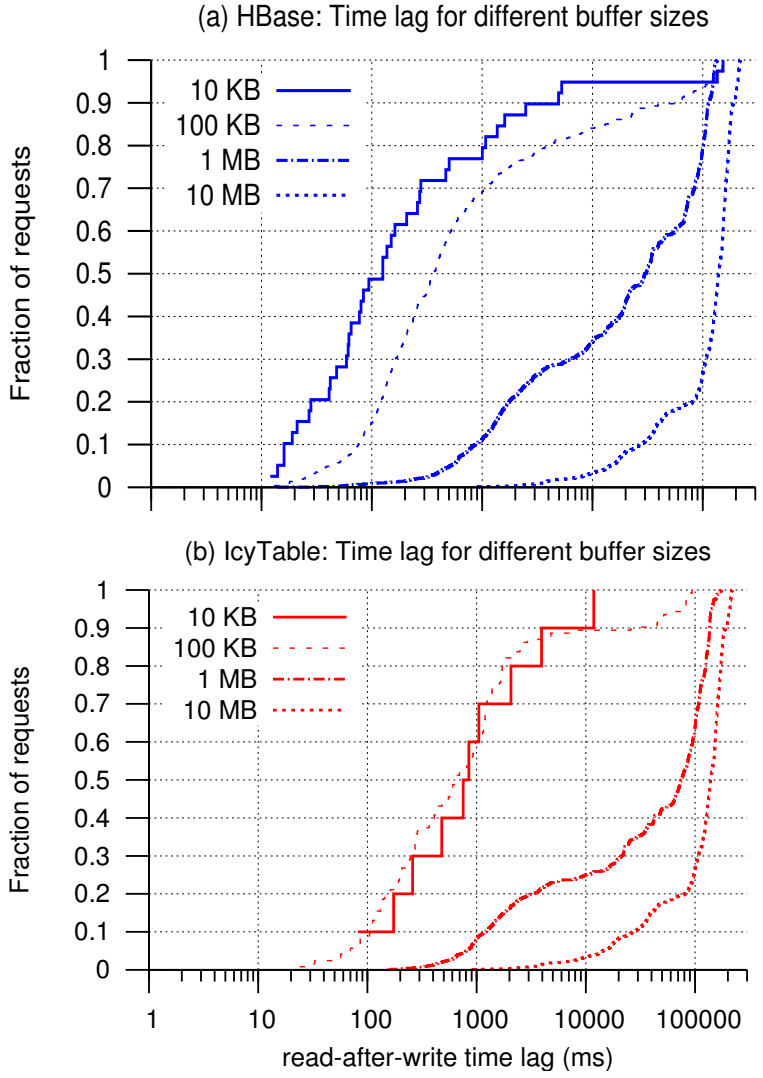




- Small batches burn excessive client CPU, limiting thruput
- Large batches saturate servers, limiting benefit of larger batch

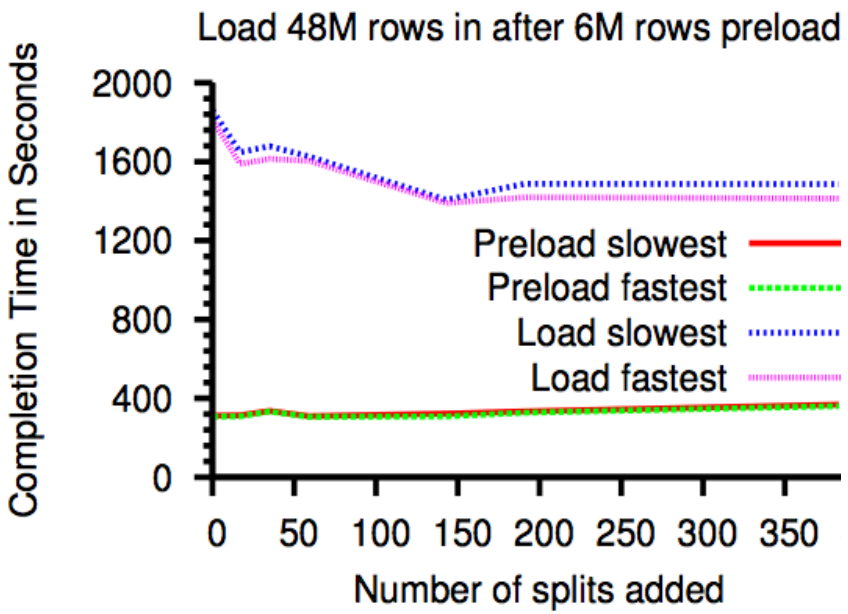
# Batch writers & eventual consistency

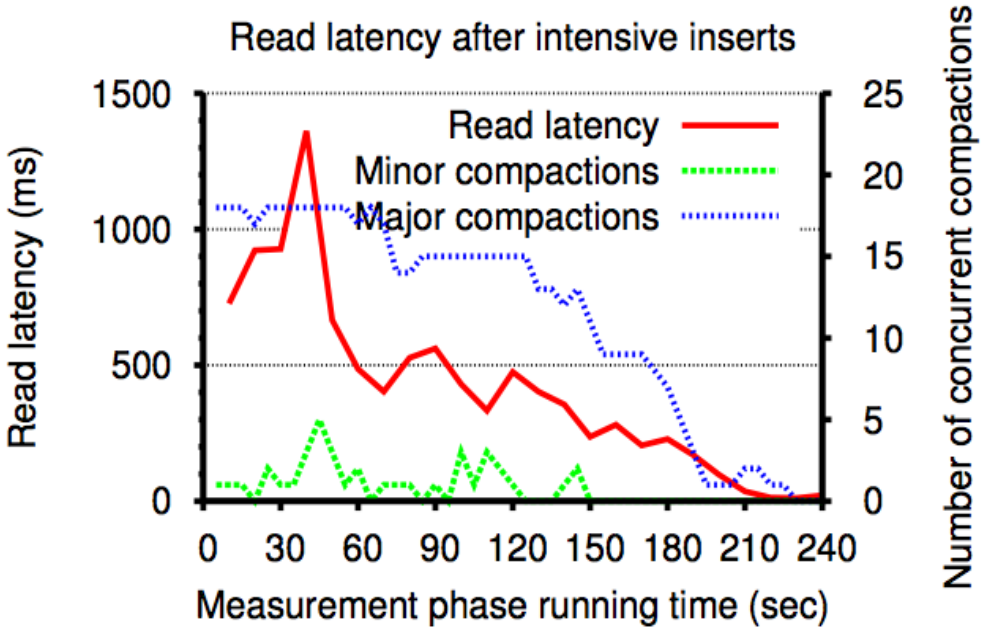



- Deferred write wins, but visible latency can be 100 secs

# Pre- (and post-) Tablet Splitting




- 6 servers
  - Per server: Preload 1M rows; Load 8M rows; Measure @100 ops/s
- 20% faster load if pre-split
  - post-load rebalancing hurts

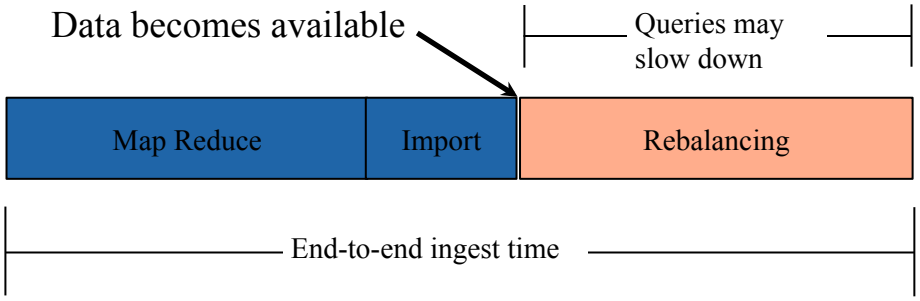
***for minutes***

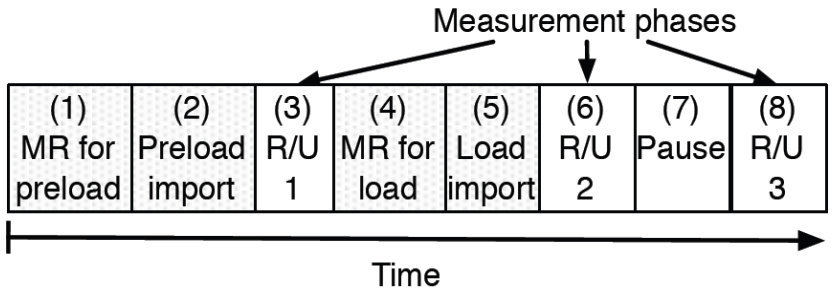
| Phase | Workload |
|---|---|
| Pre-load | Pre-load 6M rows in range $[0, 12B]$ |
| Pre-split | Pre-split tablet $[0, 72M]$ evenly |
| Load | Load 48M rows in range $[0, 72M]$ |
| Measurement 1 | Half update and half read for 4 minutes with 600 ops/s target |
| Sleep | Sleep for 5 minutes |
| Measurement 2 | Same as measurement one |

**Table 2 – Different phases in pre-split experiment**

# Improving Ingest Speed: Bulk Load

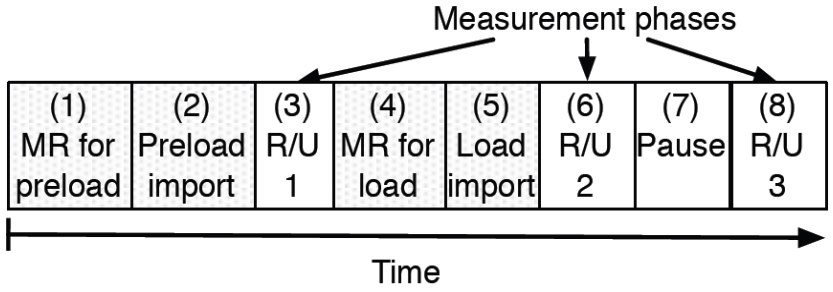
- Faster ingest is ***format*** with MapReduce, ingest/import with bulk load, ***rebalance*** during measurement phase
- Test: preload, monitor/measure, format bulk, bulk load, monitor/measure, sleep 5 minutes, monitor/measure
  - Per server: Preload 1M rows; Load 8M rows; Measure @ 100 ops/s
- Import turns out to be nearly instant, but rebalancing is not
  - Load 48M rows one at a time: 1400-1600 secs, 23-26 mins
  - Bulk load, including formatting time: 5-12 mins (***2-5X faster***)

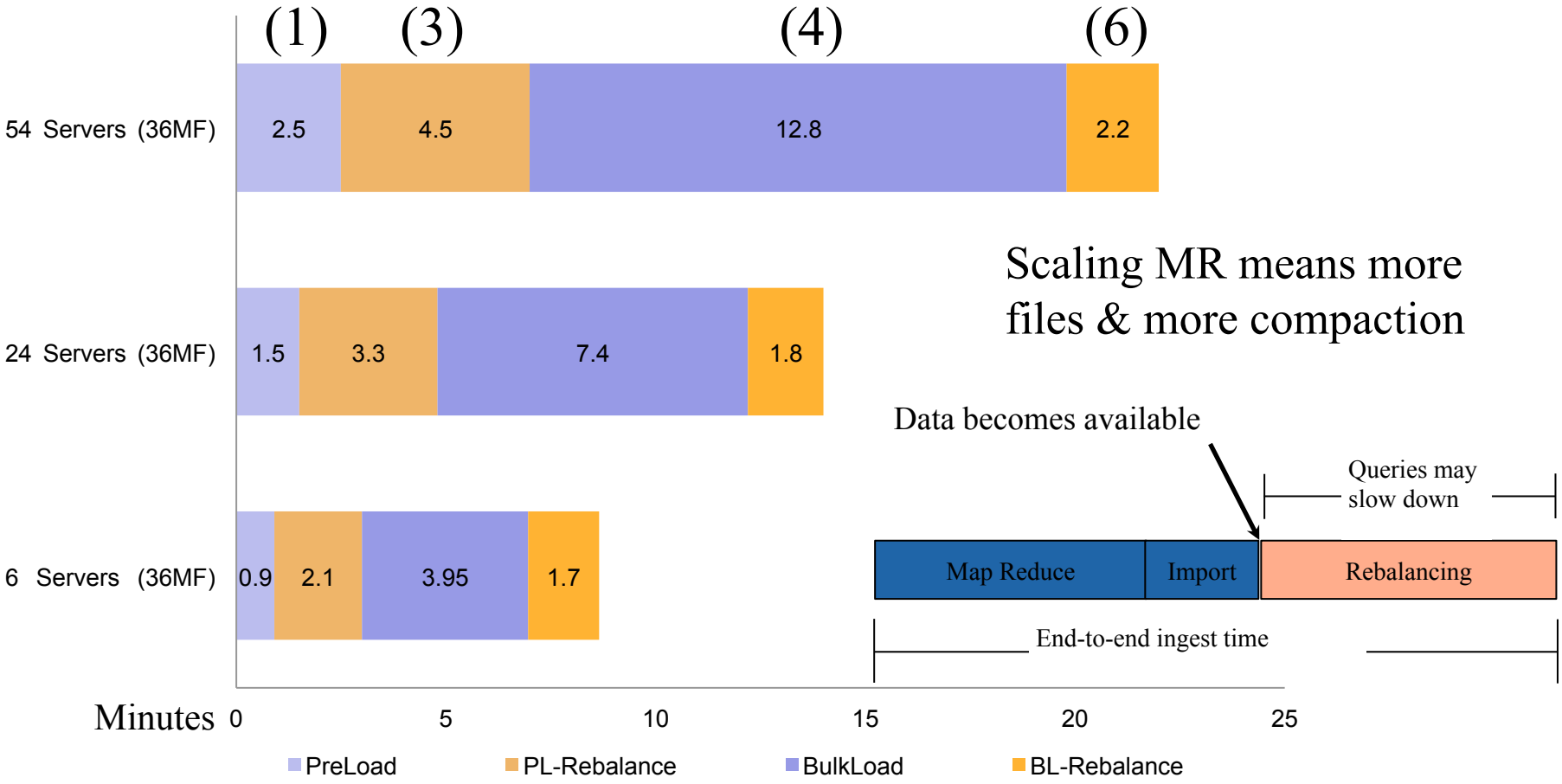


# Scaling & bulk loading

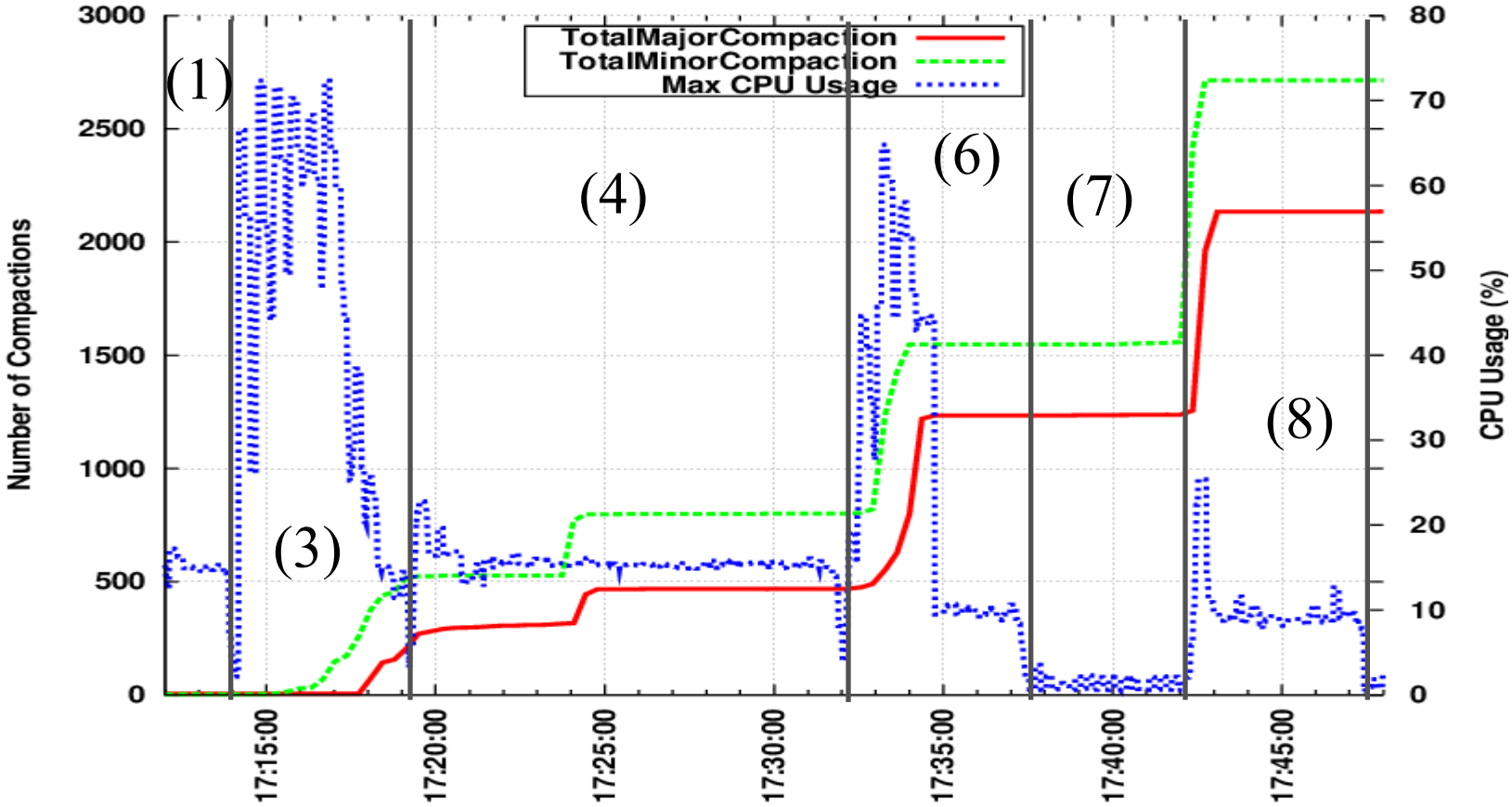
- 1/8M rows per server
- DoD BigTable



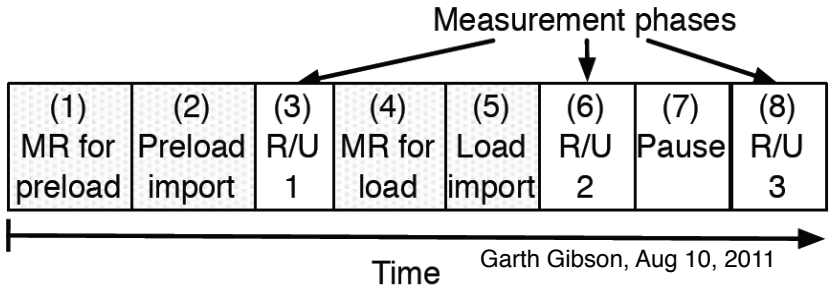




Scaling MR means more files & more compaction

# Rebalancing Timeline (54 Servers/36 MapFiles)



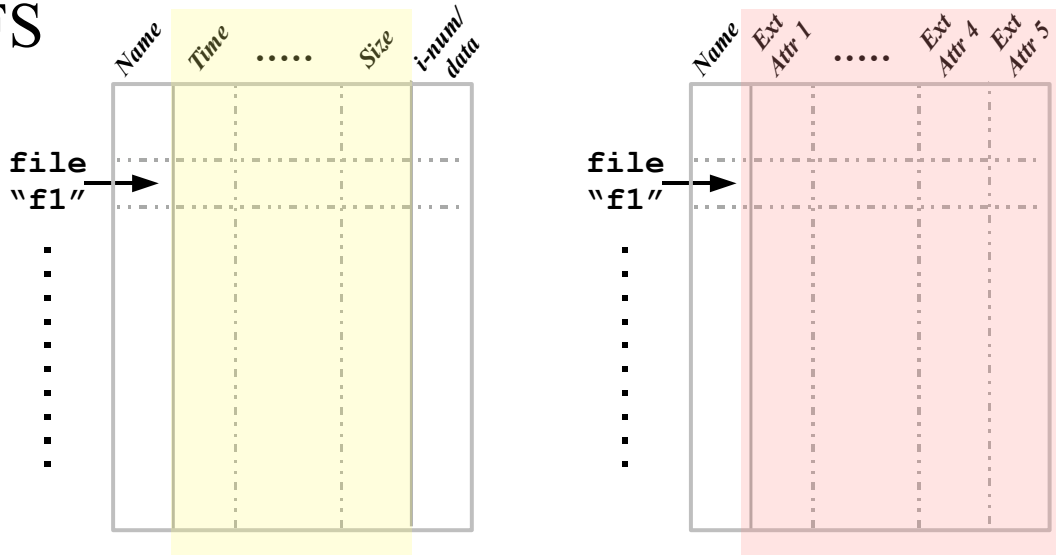

- Phase 1 rebalancing starts late
- Too much rebalancing work



# Next Up: Cloud DB inside File System?

- HDFS/GFS metadata limiting Cloud DB performance
  - Bounded number of files & metadata operations per second
  - Yahoo! Federation: many HDFS on different volumes
  - Colossus rumors: BigTable inside GFS?
- Try CloudDB inside HDFS
  - Namespace (directories)
    - How ordered, listed, clustered, (re)balanced
    - Rename transactions
  - Distr'd block allocation
  - Managing complexity of repair process

A single directory is "split" into multiple, each indexed on the same file name but with different sets of desired attributes.




***- Data contents can be embedded in the directory.***
***- Directory can be stored as a distributed B-tree for range-based queries.***